Session 304

Extensible Kernel Networking Services

Laurent Dumont
Core OS Networking

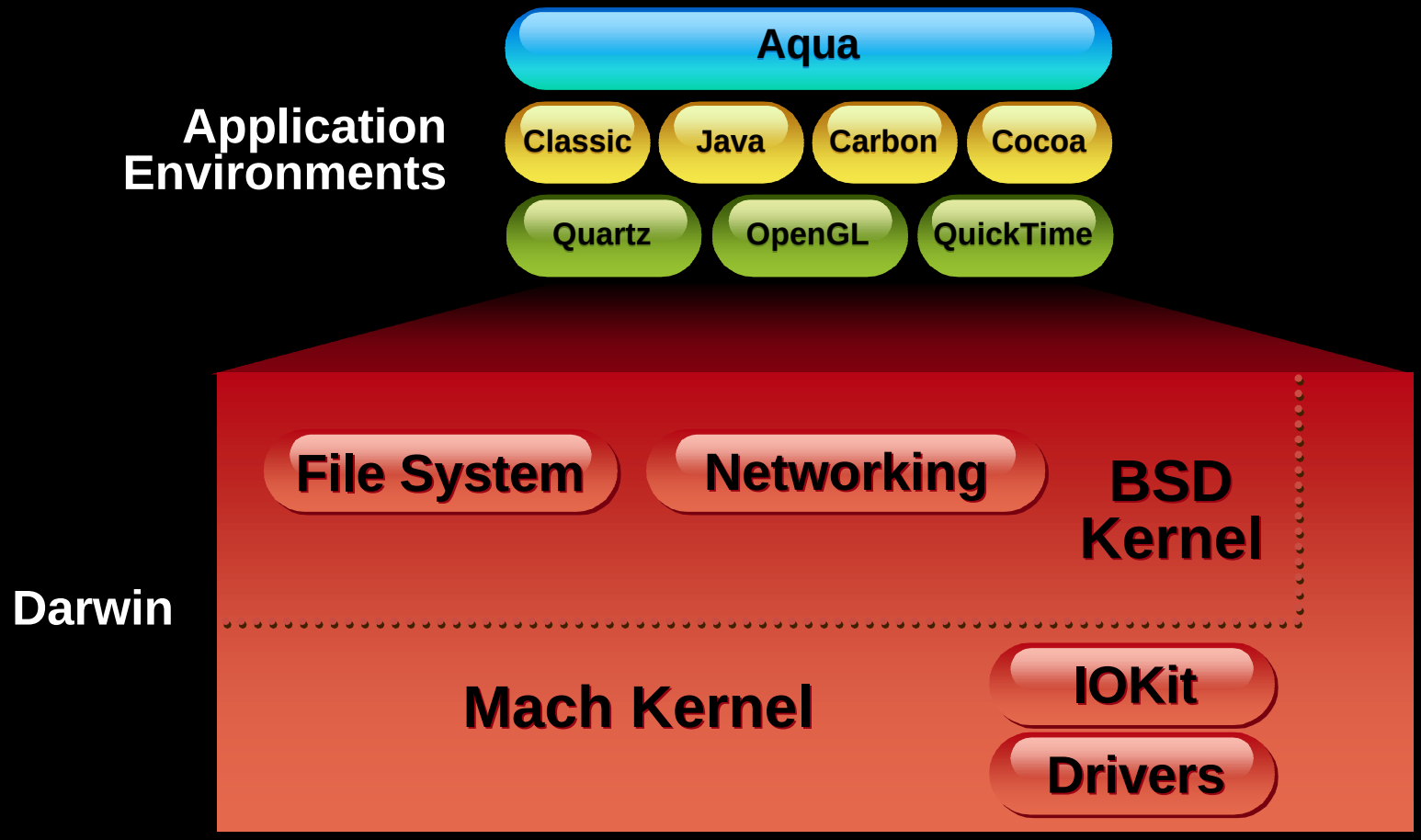
Introduction

- Mac OS X networking architecture is extensible
- The architecture allows to add:
  - Firewalls
  - VPN
  - Content filters
  - Network drivers

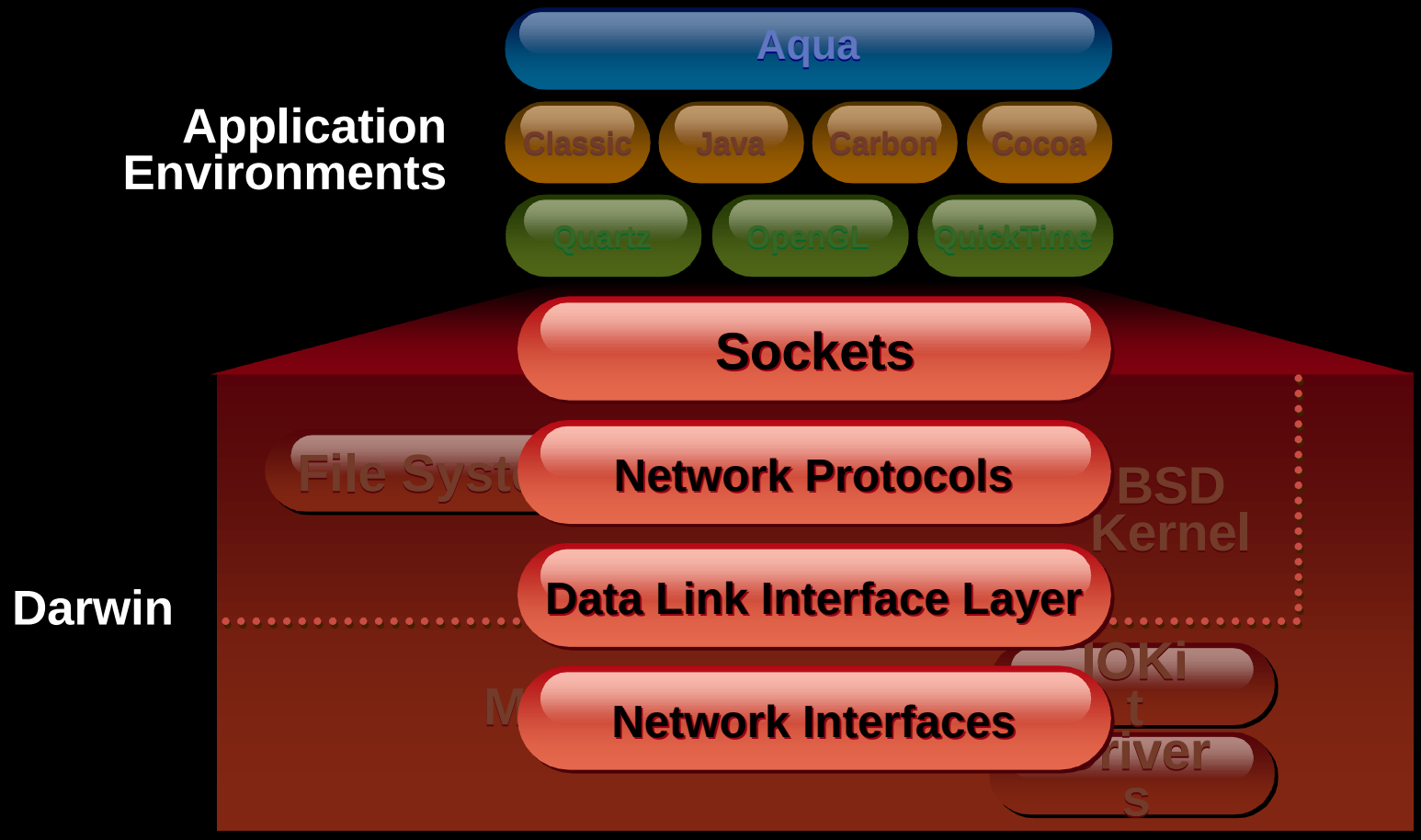
What You'll Learn

- The network kernel architecture
- How to filter/intercept packets
- How to add network interfaces and drivers
- Some important tips

Networking in Darwin



Networking in Darwin



Networking in Darwin

- Currently based on FreeBSD 3.2
- Robust and proven implementation
- Apple enhancements
  - Multithreaded and MP savvy kernel
  - Tuned network buffer allocation
  - Data Link Interface Layer (DLIL)
  - Extensible

Network Kernel Extensions

NKE

- Add extensibility to kernel networking
- Filter NKE can see, modify, inject and drop packets
- Dynamic loading of NKE, no need to reboot
- Running in the kernel comes with liabilities

Network Kernel Extensions

Proceed with caution

- Binary compatibility is not guaranteed in the future
- No real API, the Networking subsystem is wide open
- Internal structures and implementation details are exposed
- Darwin is not BSD: Darwin is based on BSD

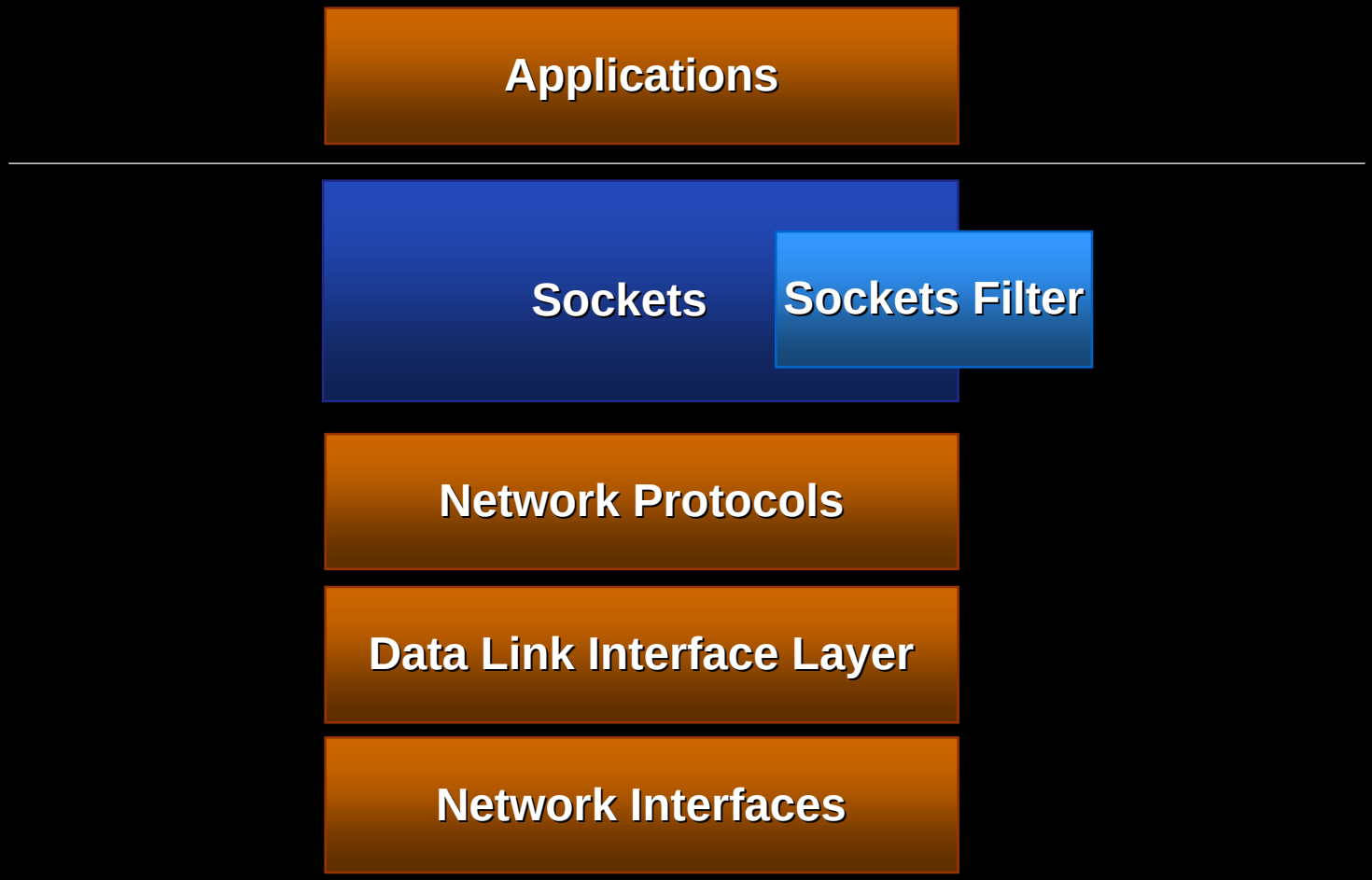
NKE Dynamic Loading

- Kernel extension mechanism provided by IOKit
- A kernel module has two entry points
  - Start routine
  - Stop routine
- Apple Kernel extensions are found in /System/Library/Extensions
- NKEs are not automatically loaded
- Command line tools
  - kextload, kmodstat, kmodunload

Types of NKE

- Sockets filters
- Network protocols
- Data Link Interface Layer filters
- Network interfaces

Sockets Layer



Sockets

- Sockets is “the” API for networking
- Sockets is the glue between applications and network protocols
- Sits above network protocols
- A socket is a kind of file structure
- A socket has pair of packet queues for incoming and outgoing packets
- Darwin sockets have plug-ins

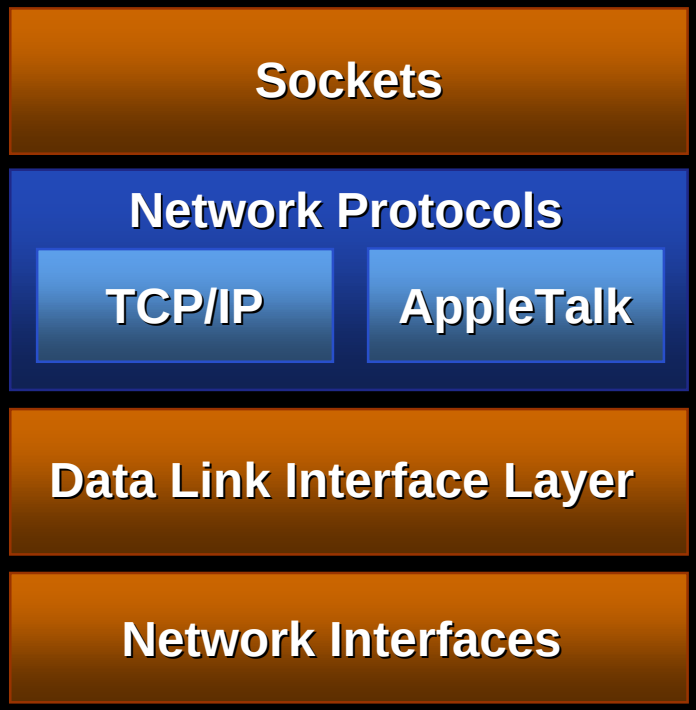
Sockets Filter NKE

- Intercept
  - Socket API calls from application
  - Socket buffers calls from protocol
- Global or programmatic
- Register your nke_handle with Apple
- Example: Content filters

Socket NKE Usage Tracking

- No built-in reference tracking
- Keep track of where your filter is inserted
- Live unload only after you remove all refs to the NKE
- NKE may refuse to unload

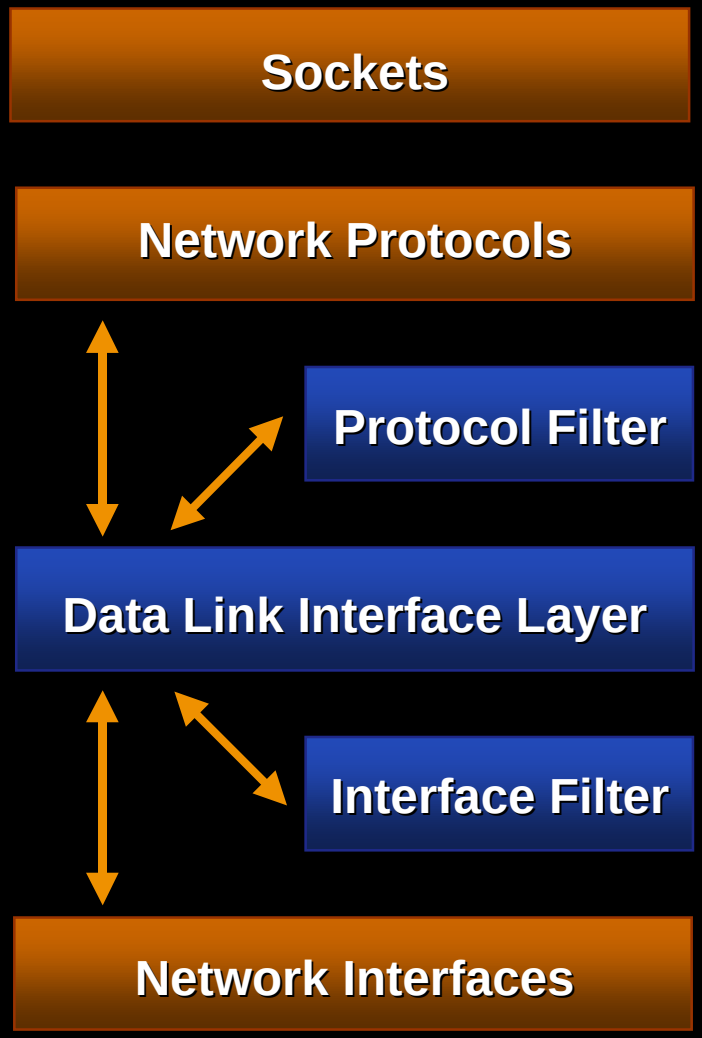
Network Protocols



Network Protocols

- A domain defines a protocol family
- A protocol family has set of protocol handlers
- Example TCP/IP:
  - Protocol family: PF_INET
  - Protocol handlers: TCP, UDP, raw IP
- Extensible: Add you own protocol

Data Link Interface Layer



Data Link Interface Layer

- Why DLIL?
  - Extensible
  - Dynamic
- DLIL is central meeting point for:
  - Protocols
  - Data link filters
  - Network interfaces

Data Link Interface Layer

- An interface attaches to DLIL to become available
- Protocols attach to some interfaces using DLIL
- The dl_tag identifies protocol attachment to an interface
- DLIL interface family per type of interface to handle framing
- DLIL protocol module (arp)

DLIL Filters

- DLIL protocol filters
  - Between family and network protocol
  - See protocol datagrams for an interface
- DLIL interface filters
  - Between an interface and family
  - See full frames for all network protocols
- For VPN and firewalls

Network Interfaces

- One ifnet structure per interface
- IOKit based
  - Ethernet
- Non-IOKit for pseudo-devices
  - PPP
  - VPN

Berkeley Packet Filter (bpf)

- Standard way to analyze network traffic for BSD
- Not really a filter, more like a “tap”
- tcpdump command rely on bpf
- Lower layer than DLIL (Promiscuous mode)
- Standard hooks for IOKit network interfaces
- Non-IOKit interfaces need to support bpf hooks

mbufs

- Memory buffers to hold network data
- Logical pointers to data (no more copies)
- Use functions to access mbufs
  - Avoid macros if possible
- Allocation can and will fail
- Use M_DON'TWAIT on input: Do not block

Kernel Events

- PF_SYSTEM sockets protocol
- Reports events from kernel to user space
- Low bandwidth
- Use by system configuration
- You can add your own events

Raw Device Access

- Sockets protocol for Network *DRiVer*
  - Not classic ndrv (native driver)
- Provides access to raw packets
- Enable user space protocol stacks
- SharedIP is an example

Funnels and NKE

Funnels

- Funnel: Mechanism to run code in mutual exclusion
- Darwin BSD has two funnels for MP performance
- Network funnel for networking
  - Sockets system calls
  - DLIL sets network funnel on input
- Kernel funnel for the rest of BSD

Funnels and NKE

When to use them?

- When to set the network funnel?
  - Module start and stop
  - Timeout function
  - New kernel thread
- When to switch funnel?
  - File system and TTY
  - Process management
- Funnel switch is a preemption point
- splnet is a no-op

NKE Control

- Control NKE from process
- NKEMgr
  - Not loaded by default
  - Will be subsumed into PF_SYSTEM
- Character device
- Access though ioctl or socket options

VPN

- Several ways to implement VPN
  - DLIL filters
  - Pseudo device
- KAME IPSec is coming to X
  - Build your own Darwin kernel with IPSec
  - Talk to us if you're interested

Summary: Rules for NKE

- Declare your dependencies
- Keep track of your resources and usage
- Check for NULL pointers
- Do not block on input or fast path
- Know your split funnels
- Be aware of binary compatibility

Future

- IPSec
- IPv6
- PPP extensibility
- NKE control via sockets API
- Finer grain locking

Additional Resources

Mac OS X
http://developer.apple.com/macosx

Darwin
http://www.publicsource.apple.com

Darwin information
http://darwinfo.org

FreeBSD
http://www.freebsd.org

Resources

TCP/IP Illustrated Volume 2
by Wright & Stevens

The Design and Implementation of the 4.4 BSD OS
by McKusick

Inside Mac OS X: Network Kernel Extensions
developer.apple.com

Kernel Extensions Tutorials
developer.apple.com

What Can We Do for You?

- What system services do you need from us to help you bring your solution on Mac OS X?

Roadmap

300 Networking Overview	Room A1 **Tues., 9:00am**
303 Network Configuration & Mobility	Room A1 **Tues., 5:00pm**
FF107 Feedback Forum: Networking	Room J1 **Thurs., 10:30am**

Who to Contact

Vincent Lubet
Core OS Networking Manager
vlubet@apple.com

Laurent Dumont
Core OS Networking
laurent@apple.com

Q&A

Thomas Weyer
Network & Communications Technology Manager
weyer@apple.com

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001